WEST Search History

DATE: Thursday, December 04, 2003

Set Name side by side	Query	Hit Count	Set Name result set
DB=USPT,PGPB; PLUR=NO; OP=ADJ			
L15	pack$3 with L14	2	L15
L14	byte$1 with L13	47	L14
L13	store with interleav$3	771	L13
L12	byte$1 with L11	18	L12
L11	load with interleav$3	479	L11
L10	6058408.pn.	1	L10
L9	interleav$3 and L8	36	L9
L8	l6 or L7	292	L8
L7	(4992934| 5269007| 5367705| 5574928| 5721892| 5734874| 5752271| 5768609| 5778241| 5812147| 5852726| 5864713| 5896543| 5913054| 5918252| 5983256)![uref]	278	L7
L6	(4992934| 5269007| 5367705| 5574928| 5721892| 5734874| 5752271| 5768609| 5778241| 5812147| 5852726| 5864713| 5896543| 5913054| 5918252| 5983256)![pn]	16	L6
DB=EPAB; PLUR=NO; OP=ADJ			
L5	load and L4	3	L5
L4	fleck.in. and siemens.as.	27	L4
DB=USPT; PLUR=NO; OP=ADJ			
L3	fleck.in. and load and packed	3	L3
DB=DWPI; PLUR=NO; OP=ADJ			
L2	fleck.in. and load and packed	0	L2
DB=EPAB; PLUR=NO; OP=ADJ			
L1	fleck.in. and load and packed	0	L1

END OF SEARCH HISTORY

WEST

Generate Collection | Print

L3: Entry 1 of 3 File: USPT Jul 10, 2001

US-PAT-NO: 6260137
DOCUMENT-IDENTIFIER: US 6260137 B1

TITLE: Data processing unit with digital signal processing capabilities

DATE-ISSUED: July 10, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Fleck; Rod G.	Mountain View	CA		
Martin; Daniel	Mountain View	CA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Siemens Aktiengesellschaft	Munich			DE	03

APPL-NO: 08/ 928764 [PALM]
DATE FILED: September 12, 1997

INT-CL: [07] G06 F 9/315

US-CL-ISSUED: 712/225; 712/224
US-CL-CURRENT: 712/225; 712/224

FIELD-OF-SEARCH: 395/393, 395/588, 395/598, 395/800.13, 395/800.22, 395/800.23, 395/800.24, 395/800.35, 395/800.36, 395/800.41, 395/380, 395/306, 395/316, 712/217, 712/241, 712/248, 712/1-43, 712/2T, 712/224, 712/225, 710/9, 711/1-6, 711/1T

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected | Search ALL

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	4992934	February 1991	Portanova et al.	712/209
☐	5269007	December 1993	Hanawa	712/218
☐	5367705	November 1994	Sites	710/21
☐	5574928	November 1996	White	712/23
☐	5721892	February 1998	Peleg	712/221
☐	5734874	March 1998	Van Hook	345/513
☐	5752271	May 1998	Yung	712/23
☐	5768609	June 1998	Gove et al.	712/11
☐	5778241	July 1998	Bindloss et al.	712/20
☐	5812147	September 1998	Van Hook	345/511
☐	5852726	December 1998	Lin et al.	712/200
☐	5864713	January 1999	Terry	395/872
☐	5896543	April 1999	Garde	712/35
☐	5913054	June 1999	Mallick et al.	711/200
☐	5918252	June 1999	Chen et al.	711/217
☐	5983256	November 1999	Peleg et al.	708/523

FOREIGN PATENT DOCUMENTS

FOREIGN-PAT-NO	PUBN-DATE	COUNTRY	US-CL
0 473 805 A1	September 1990	EP	
0 679 991 A1	November 1995	EP	
WO 96/17291	June 1996	WO	

OTHER PUBLICATIONS

Sun Microsystems, "Ultrasparc-I,-II User's manual", pp. 190-234, [retrieved on Jul. 7, 1999]. Retieved from the Internet:<URL: http://www.sun.com/microelectronics/UltraSPARC-II/; $sessionid$2EHV1ZQAAFGW5AMUVFZE5YQ.*
Sun Microsystems, "Ultrasparc-IIi User's manual", pp.127-133, Oct. 1997.

ART-UNIT: 282

PRIMARY-EXAMINER: Niebling; John F.

ASSISTANT-EXAMINER: Whitmore; Stacy

ABSTRACT:

The present invention relates to a data processing unit comprising a register file, a register load and store buffer connected to the register file, a single memory, and a bus having at least first and second word lines to form a double word wide bus coupling the register load and store buffer with said single memory. The register file at least two sets of registers whereby the first set of registers can be coupled with one of the word lines and the second set of registers can be coupled with the respective other word lines, a load and store control unit for transferring data from or to the memory.

24 Claims, 9 Drawing figures

WEST

Generate Collection Print

L3: Entry 1 of 3 File: USPT Jul 10, 2001

DOCUMENT-IDENTIFIER: US 6260137 B1
TITLE: Data processing unit with digital signal processing capabilities

Abstract Text (1):
The present invention relates to a data processing unit comprising a register file, a register load and store buffer connected to the register file, a single memory, and a bus having at least first and second word lines to form a double word wide bus coupling the register load and store buffer with said single memory. The register file at least two sets of registers whereby the first set of registers can be coupled with one of the word lines and the second set of registers can be coupled with the respective other word lines, a load and store control unit for transferring data from or to the memory.

INVENTOR (1):
Fleck; Rod G.

Brief Summary Text (8):
This object is accomplished by a Data processing unit comprising a register file, a register load and store buffer connected to the register file, a single memory, and a bus having at least first and second word lines to form a double word wide bus coupling the register load and store buffer with said single memory. The register file at least two sets of registers whereby the first set of registers can be coupled with one of the word lines and the second set of registers can be coupled with the respective other word lines, a load and store control unit for transferring data from or to the memory.

Brief Summary Text (9):
In one embodiment, the load and store control unit has means to load or store two consecutive words in parallel from or to said memory to or from the first and second set of registers. In another embodiment, one word from the memory can be split into two half-words which are then stored in a first register from the first set of registers and in a second register from the second set of registers. The half-words can be stored into one half of a register and the other half of the register can be filled up with zeros or sign-filled.

Brief Summary Text (10):
In a further embodiment the bus has a plurality of word lines to form a plurality-word wide bus and the register file has a plurality of sets of registers whereby each set of registers is coupled with one of word lines of said plurality of word lines. For example, in a 64 bit data processing unit, two 32 bit half-words or four 16 quarter-words can be accessed during one single cycle. The load and store control unit of the data processing can therefore have means to load or store a plurality of consecutive words in parallel from or to said memory to or from said plurality of sets of registers. These means allow to couple any register of any set of registers with any location within the memory.

Brief Summary Text (11):
In a further embodiment the load and store control unit of the data processing unit can have means to load one word from said memory and to split it into a plurality of partial-words, each partial word is stored in one of said registers of each set of registers, respectively.

Detailed Description Text (3):

For coupling the register file 2 with the memory 1, a buffer/select logic 2a is provided. In this embodiment, numeral 2b indicates the registers. 16 registers D0 to D15 are provided, whereby each register has a bit width of a word which has, for example, 32 bits. The registers are organized in two groups, even and odd registers. The registers in this example are data register but can be either address or data registers. A second set of registers can be provided in the same way for address registers. The bus between the memory unit 1 and the buffer/select logic 2a is 64 bits wide thereby two consecutive words in the memory 1 can be addressed. A load/store control unit 2d addresses the memory unit 1 and selects the respective registers 2b during a transfer from the register file 2 to the memory unit 1 or vice versa. The register file 2 comprises furthermore a second buffer/select logic 2c coupling a plurality of execution units 4, 5, and 6 thereto. A second bus 3 is provided as a link between the buffer/select logic 2c and the execution units 4, 5, and 6. Through the respective buffer/select logic 2a or 2c at least two registers, one in each group, for example, an even and an odd register, can be accessed at the same time.

Detailed Description Text (6):
The special instructions provide a "load double word to a register"-instruction. The double word is loaded from the memory to the multiplexer units 8 and 9 through the data output lines 1a and 1d. In this mode units 12 and 13 operate as multiplexers coupling the data lines 1a with the odd registers or with even registers and the data lines 1d with the even registers or the odd registers, respectively. The data processing unit can have a special selecting unit allowing to select in this instruction any register in each group. A simplified embodiment selects only one register and the second register is automatically the register adjacent to the selected one. For example, if the even register D4 is the selected, the adjacent odd register would be register D5 or if the odd register D7 would be selected, the adjacent even register would be D6. The double word in the memory can be located at aligned addresses, for example word 1e, and consecutive word 1f, or it can be accessed at unaligned addresses, such as word 1f and consecutive word 1g. The multiplexer 7, 8,9, and 10 align the respective data and distribute them to the respective registers or memory cells.

Detailed Description Text (8):
A second type of instruction which can be executed according to the present invention is a so called "load two half-words (packed)"-instruction. With this instruction one word from either data lines 1a or 1d is loaded and split into half-words by units 8 or 9 placed in the respective lower halves of a word. Optionally units 12 and 13 can either sign-extend or zero-extend the respective half-words to words. In other words, in this embodiment, the 16 bit half-words are extended to 32 bits. Unit 8 or unit 9 splits the word received from lines 1a or 1d into two half-words and distributes them through units 12 and 13 to the lower halves of the respective even and odd registers. In units 12 and 13 these half-words can be extended to words either by filling the upper halves with zeros or by sign extending the upper halves. If the sign of a half-word is negative the upper halves of the respective register is filled up with "1" otherwise with "0". If units 12 and 13 are deactivated the half-words are stored into the lower halves of the respective even and odd registers without changing their upper halves. In a simplified version the least significant memory half-word is always stored into an even register and the most significant half-word is stored into an odd register adjacent to the even register.

Detailed Description Text (9):
A third type of instruction which can be executed according to the present invention is a so called "load two signed fractions"-instruction. With this instruction one word from either data lines 1a or 1d is loaded and split into half-words by units 8 or 9 placed in the upper halves of a respective word. Optionally units 12 and 13 can zero-extend the respective half-words to words. Unit 8 or unit 9 splits the word received from lines 1a or 1d into two half-words representing the upper and lower half of the word and distributes them through units 12 and 13 to the upper halves of the respective even and odd registers. In units 12 and 13 these half-words can be extended to words by filling the lower halves with "0". If units 12 and 13 are deactivated the half-words are stored into the upper halves of the respective even and odd registers without changing their lower halves. In a simplified version the least significant memory half-word is always stored into an even register and the most significant half-word is stored into an odd register adjacent to the even register.

Detailed Description Text (10):
A fourth type of instruction which can be executed according to the present invention is a so called "store two half-words (packed)"-instruction. With this instruction the lower half-words of an even and an odd register are fed to either concatenating unit 11 or 14. The two half-words are combined to one word and the stored in the memory unit 1 through multiplexer 7 or 10 and either data input lines 1b or 1c.

Detailed Description Text (12):
Finally a sixth type of instruction which can be executed according to the present invention is a so called "store double word from data registers"-instruction. With this instruction the content of an even and an odd register are fed to either multiplexer units 7 or 10 and stored in the memory unit through data input lines 1b and 1c. This instruction works in the same way as a "load double word to a register"-instruction described above. Units 7 and 10 operate as multiplexers distributing the content of each register to either data input lines 1b or 1c. Units 11 and 14 are deactivated so that units 7 and 10 each receive the full word stored in an even or odd register at their inputs.

Detailed Description Text (13):
This principle of arranging the memory and the register file can be easily extended. For example, four different sets of register can be provided and the addressing of the memory can be extended by a four word wide bus, allowing to load and store four consecutive words at a time.

Detailed Description Text (16):
FIG. 4 shows a second type of packed arithmetic or logical operations. Three registers 20, 25 and 26 is divided into four parts. In this embodiment, each part contains 8 bit. Four operator units 21, 22, 23, and 24 are provided and associated to each 8 bit part of registers 20, 25 and 26. The four parts of registers 20 and 25 provide the input values for each operator unit 21, 22, 23, and 24, whereas the output signals of each operator unit 21, 22, 23, and 24 are fed to the respective parts of register 26.

Detailed Description Text (19):
This so called packed arithmetic or logical instructions partition, in this embodiment, a 32 bit word into several identical objects, which can then be fetched, stored, and operated on in parallel. These instructions, in particular, allow the full exploitation of the 32 bit word of the data processing unit according to the present invention in DSP applications.

Detailed Description Text (20):
In this embodiment two packed formats can be implemented. The first format divides the 32 bit word into two 16 bit half-word values. The second packed format divides the 32 bit word into four 8 bit (byte) values.

Detailed Description Text (21):
The loading and storing of packed values into data or address registers is supported by the respective load and store instructions described above. The packed objects can then be manipulated in parallel by a set of special packed arithmetic instructions that perform such arithmetic operations as addition, subtraction, multiplication, division, etc. For example a multiply instruction performs two, 16 bit multiplication's in parallel as shown in FIG. 5.

Detailed Description Text (23):
Addition is performed on individual packed bytes or half-words using the respective addition instructions and they can be extended by a saturation unit 44 which ignores overflow or underflow within individual bytes or half-words. The saturation unit 44 provides each addition with a function that saturates individual bytes or half-words to the most positive value on individual overflow or to the most negative value on individual underflow. For example, compare unit 41 can compare the content of result register 42 with a predefined saturation value. If the content is greater than a predefined positive/negative saturation value, this is indicated to saturation unit 44 and saturation unit 44 sets the content of result register 42 to the respective positive or negative saturation value. Saturation can be provided to a variety of arithmetic instructions.

Detailed Description Text (26):
A circular buffer control unit 32 is coupled with these registers 31a, 31b, and 31c. A load/store control unit for the circular buffer 33 is coupled with this control unit 32 and with the memory 1 and the register file. It also has access to the buffer storing means 31. The instruction execution unit of the CPU is indicated by numeral 34 and receives certain control inputs as will be explained later.

Detailed Description Text (28):
FIG. 6 shows such a circular buffer consisting of memory cells b1, b2, . . . b8. If the circular buffer control unit starts accessing the buffer beginning with a starting index of "0", the first two cells b1 and b2 and the consecutive cells are accessed aligned, no further control action is necessary. If a starting index of, for example "1" is used, or the offset is an odd number a double word access beginning at word b8 must access word b1 as the second word. As word b1 is not consecutively stored in regard to word b8, load/store control unit 33 issues a second instruction into the instruction execution unit 34 to access word b8 during a first cycle and word b1 during the following cycle. Only in this case two access cycles are necessary to load or store data which cross the boundary of the circular buffer. As circular buffers are usually large such accesses are very rare compared to "normal" non-boundary-crossing access.

Detailed Description Text (33):
FIG. 9 shows a block diagram showing an example of a configuration of a data handling unit according to the present invention to perform a FIR filter function. A memory 1 contains Data 0 to Data N-1 and coefficients COE 0 to COE N-1. The memory is addressed by the address register file 45 which contain respective pointers and which is coupled with a load/store address arithmetic. The memory 1 is also connected through a 64 bit bus with the data register file 2 containing actual coefficients and data which are calculated. The data processing unit comprises a plurality of buses 47, 48, 49 and 50 which handle the different data for execution in the different arithmetic units. Two multipliers 51 and 52 are provided to execute two multiplication's in parallel whose inputs are coupled with the data register file through bus 47. Furthermore two 16 bit adders 53 and 54 are provided which are coupled through bus 50 with the results of the multipliers 51 and 52. Bus 48 is coupled to the outputs of adders 53 and 54. Two additional adders 55 and 56 are provided whose inputs are coupled with bus 48 and whose outputs are coupled to bus 49. Bus 47 and therefore data register file 2 is coupled through several lines with busses 48 and 49. Bus 50 and bus 49 are additionally coupled with bus 48.

CLAIMS:

1. Data processing unit comprising:

a register file with a plurality of word-wide registers, whereby a word having a predefined number of bits,

a register load and store buffer connected to said register file,

a memory,

a bus having at least first and second word lines to form a double word wide bus coupling said register load and store buffer with said memory, whereby said register file has at least two sets of registers,

coupling means, so that said first set of registers can be coupled with one of said word lines and said second set of registers can be coupled with the respective other word lines,

a load and store control unit for transferring data from or to said memory, wherein said load and store control unit is configured to, in response to a single instruction for the data processing unit, load one word from said memory and to split it into two half-words which are stored in one half of a first register from said first set of registers and in a corresponding half of a second register from said second set of registers. respectively.

2. Data processing unit according to claim 1, wherein said load and store control unit has means to load a first half-word from a first register of said first set of registers and a second half-word from a second register from said second set of registers and to concatenate both half-words to a single word and to store said word in said memory via said data bus.

14. Data processing unit according to claim 1, wherein said load and store control unit has means to load or store two consecutive words in parallel from or to said memory to or from said first and second set of registers.

15. Data processing unit according to claim 1, wherein said load and store control unit has means to load or store two consecutive words in parallel from or to said memory to or from said first and second set of registers.

16. Data processing unit according to claim 1, wherein said one half of said first register is the lower half of said first register, whereby said corresponding half of said second register is therefore the lower half of said second register, and wherein said load and store control unit is further configured to sign fill the upper half of each of said first and second registers in response to said single instruction.

17. Data processing unit according to claim 1, wherein said load and store control unit is configured to fill the other half of each of said first and second registers with zeros.

18. Data processing unit comprising:

a register file with a plurality of word-wide registers, whereby a word having a predefined number of bits,

a register load and store buffer connected to said register file,

a memory,

a bus having at least first and second word lines to form a double word wide bus coupling said register load and store buffer with said memory, whereby said register file has at least two sets of registers,

coupling means, so that said first set of registers can be coupled with one of said word lines and said second set of registers can be coupled with the respective other word lines,

a load and store control unit for transferring data from or to said memory, wherein said load and store control unit has means to load one word from said memory and to split it into two half-words which are stored in a first register from said first set of registers and in a second register from said second set of registers, wherein said load and store control unit further comprises means to load said half-words into a lower half of a register and to sign fill the upper half of said register.

19. Data processing unit comprising:

a register file with a plurality of word-wide registers whereby a word having a predefined number of bits,

a register load and store buffer connected to said register file,

a memory,

a bus having at least first and second word lines to form a double word wide bus coupling said register load and store buffer with said memory, whereby said register file has at least two sets of registers,

coupling means, so that said first set of registers can be coupled with one of said word lines and said second set of registers can be coupled with the respective other word lines,

a load and store control unit for transferring data from or to said memory, wherein said load and store control unit has means to load one word from said memory and to split it into a plurality of partial-words, each partial word is stored in one of said registers of each set of registers, respectively, wherein said load and store control unit further comprises means to load said partial-words into one part of a register and to fill the remaining part of said register with zeros.

20. Data processing unit comprising:

a register file comprising a plurality of sets of word-wide registers, wherein a word has a predefined number of bits,

a register load and store buffer coupled to said register file,

a memory,

a bus comprising a plurality of word lines to form an at least double word-wide bus coupling said register load and store buffer with said memory,

a logic configured to couple a first set of registers with one of said plurality of word lines and to couple a second set of registers with another of said plurality of word lines,

a load and store control unit configured to transfer data from or to said memory, wherein said load and store control unit is further configured to load one word from said memory, separate said one word into a plurality of partial words, and store said partial words into a plurality of said word-wide registers, each of said plurality of said word-wide registers storing no more than one of said partial words, and said partial words each stored at a same positional portion within its respective word-wide register, whereby gaps are created in said respective word-wide registers, the gaps being portions of said respective word-wide registers other than said same positional portion.

21. Data processing unit according to claim 20, wherein said load and store control unit is configured to execute a single instruction, for the data processing unit, that instructs the data processing unit to load said one word from said memory, separate said one word into said plurality of partial words, and store said partial words into said plurality of said word-wide registers.

22. Data processing unit according to claim 20, wherein said same positional portion of any word-wide register is a lower portion of said any word-wide register, and said load and store control unit is configured to sign fill an upper portion of each of said respective word-wide registers.

23. Data processing unit according to claim 20, wherein said load and store control unit is configured to zero fill said gaps.

24. Data processing unit comprising:

a register file comprising a plurality of sets of registers, each register being at least word wide, wherein a word has a predefined number of bits,

a register load and store buffer coupled to said register file,

a memory,

a bus comprising a plurality of word lines to form an at least double word-wide bus coupling said register load and store buffer with said memory,

a logic configured to couple a first set of registers with one of said plurality of word lines and to couple a second set of registers with another of said plurality of word lines,

a load and store control unit configured to transfer data from or to said memory,

wherein said load and store control unit is further configured execute an instruction that instructs the data processing unit to load one word from said memory, separate said one word into a plurality of partial words, and store said partial words into a plurality of said registers wherein said instruction also instructs the data processing unit to zero fill or sign fill.

WEST Search History

DATE: Thursday, December 04, 2003

Set Name side by side	Query	Hit Count	Set Name result set
DB=USPT,PGPB; PLUR=NO; OP=ADJ			
L12	byte$1 with L11	18	L12
L11	load with interleav$3	479	L11
L10	6058408.pn.	1	L10
L9	interleav$3 and L8	36	L9
L8	l6 or L7	292	L8
L7	(4992934| 5269007| 5367705| 5574928| 5721892| 5734874| 5752271| 5768609| 5778241| 5812147| 5852726| 5864713| 5896543| 5913054| 5918252| 5983256)![uref]	278	L7
L6	(4992934| 5269007| 5367705| 5574928| 5721892| 5734874| 5752271| 5768609| 5778241| 5812147| 5852726| 5864713| 5896543| 5913054| 5918252| 5983256)![pn]	16	L6
DB=EPAB; PLUR=NO; OP=ADJ			
L5	load and L4	3	L5
L4	fleck.in. and siemens.as.	27	L4
DB=USPT; PLUR=NO; OP=ADJ			
L3	fleck.in. and load and packed	3	L3
DB=DWPI; PLUR=NO; OP=ADJ			
L2	fleck.in. and load and packed	0	L2
DB=EPAB; PLUR=NO; OP=ADJ			
L1	fleck.in. and load and packed	0	L1

END OF SEARCH HISTORY

WEST

End of Result Set

Generate Collection | Print

L5: Entry 3 of 3 File: EPAB Mar 17, 1999

PUB-NO: EP000902360A2
DOCUMENT-IDENTIFIER: EP 902360 A2
TITLE: Apparatus for read/write-access to registers in a central processing unit

PUBN-DATE: March 17, 1999

INVENTOR-INFORMATION:

NAME	COUNTRY
FLECK, ROD G	US
ARNOLD, ROGER D	US

ASSIGNEE-INFORMATION:

NAME	COUNTRY
SIEMENS MICROELECTRONICS INC	US

APPL-NO: EP98117255
APPL-DATE: September 11, 1998

PRIORITY-DATA: US92842897A (September 12, 1997)

INT-CL (IPC): G06 F 9/30
EUR-CL (EPC): G06F009/30; G06F009/30, G06F009/315 , G06F009/38

ABSTRACT:

CHG DATE=19990905 STATUS=O> The present invention is related to a data processing unit having a set of data registers and a set of address registers. Each register has a width of n bits. Furthermore, there are provided address load and store buffers associated with said address registers, data load and store buffers associated with said data registers and a bus having a plurality of bus lines being connected to said store buffers. A data memory unit is connected to said bus. The data registers are arranged in such a way that at least n data registers are connected in parallel to respective bus lines, n being greater than 1, and the address registers are arranged in such a way, that at least m address registers are coupled in parallel to respective bus lines, m being greater than 1. Thus, at least four registers can be accessed in parallel.

WEST

Generate Collection Print

L5: Entry 2 of 3 File: EPAB Mar 25, 1999

PUB-NO: WO009914663A2
DOCUMENT-IDENTIFIER: WO 9914663 A2
TITLE: DATA PROCESSING UNIT WITH DIGITAL SIGNAL PROCESSING CAPABILITIES

PUBN-DATE: March 25, 1999

INVENTOR-INFORMATION:

NAME	COUNTRY
FLECK, ROD G	
MARTIN, DANIEL	

ASSIGNEE-INFORMATION:

NAME	COUNTRY
SIEMENS MICROELECTRONICS INC	US

APPL-NO: US09818574
APPL-DATE: September 4, 1998

PRIORITY-DATA: US92876497A (September 12, 1997)

INT-CL (IPC): G06 F 9/30
EUR-CL (EPC): G06F009/312; G06F009/30

ABSTRACT:

CHG DATE=19990905 STATUS=C>The present invention relates to a data processing unit comprising a register file, a register load and store buffer connected to the register file, a single memory, and a bus having at least first and second word lines to form a double word wide bus coupling the register load and store buffer with said single memory. The register file has at least two sets of registers whereby the first set of registers can be coupled with one of the word lines and the second set of registers can be coupled with the respective other word lines, a load and store control unit for transferring data from or to the memory.

WEST

End of Result Set

Generate Collection | Print

L10: Entry 1 of 1 File: USPT May 2, 2000

US-PAT-NO: 6058408
DOCUMENT-IDENTIFIER: US 6058408 A
**** See image for Certificate of Correction ****

TITLE: Method and apparatus for multiplying and accumulating complex numbers in a digital filter

DATE-ISSUED: May 2, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Fischer; Stephen A.	Rancho Cordova	CA		
Mennemeier; Larry M.	Boulder Creek	CA		
Peleg; Alexander D.	Carmelia			IL
Dulong; Carole	Saratoga	CA		
Kowashi; Eiichi	Ibaraki			JP

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Intel Corporation	Santa Clara	CA			02

APPL-NO: 08/ 575778 [PALM]
DATE FILED: December 20, 1995

PARENT-CASE:
This is a continuation-in-part of application Ser. No. 08/523,211, filed on Sep. 5, 1995, that is currently abandoned.

INT-CL: [07] G06 F 17/10

US-CL-ISSUED: 708/322
US-CL-CURRENT: 708/322

FIELD-OF-SEARCH: 364/724.19-724.2, 375/232-236, 708/322-323

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected | Search ALL

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	3641736	February 1972	Amdahl et al.	364/736
☐	3711692	January 1973	Batcher	235/175
☐	3723715	March 1973	Chen et al.	235/175
☐	4161784	July 1979	Cushing et al.	364/748
☐	4344151	August 1982	White	364/754
☐	4393468	July 1983	New	364/736
☐	4418383	November 1983	Doyle et al.	364/200
☐	4498177	February 1985	Larson	371/52
☐	4707800	November 1987	Montrone et al.	364/788
☐	4771379	September 1988	Ando et al.	364/200
☐	4779218	October 1988	Jauch	364/736
☐	4989168	January 1991	Kuroda et al.	364/715.09
☐	5095457	March 1992	Jeong	364/758
☐	5111422	May 1992	Ullrich	364/750.5
☐	5187679	February 1993	Vassiliadis	364/786
☐	5222037	June 1993	Taniguchi	364/748
☐	5227994	July 1993	Mitsuharu	364/750.5
☐	5241492	August 1993	Girardeau, Jr.	364/736
☐	5243624	September 1993	Paik et al.	364/724.2
☐	5262976	November 1993	Young et al.	364/760
☐	5293558	March 1994	Narita et al.	364/752
☐	5321644	June 1994	Schibinger	366/737
☐	5325320	June 1994	Chiu	364/760
☐	5381357	January 1995	Wedgewood et al.	364/724.19
☐	5420815	May 1995	Nix et al.	364/750.5
☐	5441799	August 1995	Murakami et al.	395/800
☐	5457805	October 1995	Nakamura	395/800
☐	5473557	December 1995	Harrison et al.	364/736
☐	5487022	January 1996	Simpson et al.	364/715.04
☐	5500811	March 1996	Corry	364/724.16
☐	5506865	April 1996	Weaver, Jr.	370/335
☐	5509129	April 1996	Guttag et al.	395/379
☐	5517438	May 1996	Dao-Trong et al.	364/748
☐	5528529	June 1996	Seal	364/736
☐	5566101	October 1996	Kodra	364/724.16
☐	5576983	November 1996	Shiokawa	364/754
☐	5675526	October 1997	Peleg et al.	364/754
☐	5677862	October 1997	Peleg et al.	364/754

OTHER PUBLICATIONS

J. Shipnes, Graphics Procesing with the 88110 RISC Microprocessor, IEEE (1992), pp. 169-174.

MC88110 Second Generation RISC Microprocessor User's Manual, Motorola Inc. (1991).
Errata to MC88110 Second Generation RISC Microprocessor User's Manual, Motorola Inc. (1992), pp. 1-11.
MC88110 Programmer's Reference Guide, Motorola Inc. (1992), pp. 1-4.
i860.TM. Microprocessor Family Programmer's Reference Manual, Intel Corporation (1992), Ch. 1, 3, 8, 12.
R. B. Lee, Accelerating Multimedia With Enhanced Microprocessors, IEEE Micro (Apr. 1995), pp. 22-32.
TMS320C2x User's Guide, Texas Instruments (1993) PP. 3-2 through 3-11; 3-28 through 3-34; 4-1 through 4-22; 4-41; 4-103; 4-119 through 4-120; 4-122; 4-150 through 4-151.
L. Gwennap, New PA-RISC Processor Decodes MPEG Video, Microprocessor Report (Jan. 1994), pp. 16, 17.
SPARC Technology Business, UltraSPARC Multimedia Capabilities On-Chip Support for Real-Time Video and Advanced Graphics, Sun Microsystems (Sep. 1994).
Y. Kawakami et al., LSI Applications: A Single-Chip Digital Signal Processor for Voiceband Applications, Solid State Circuits Conference, Digest of Technical Papers; IEEE International (1980).
B. Case, Philips Hopes to Displace DSPs with VLIW, Microprocessor Report (Dec. 1994), pp. 12-18.
N. Margulis, i860 Microprocessor Architecture, McGraw-Hill, Inc. (1990) Ch. 6, 7, 8, 10, 11.
Pentium Processor User's Manual, vol. 3: Architecture and Programming Manual, Intel Corporation (1993), Ch. 1, 3, 4, 6, 8, and 18.
Desktop Video Data Handbook, North American Philips Corporation (1993), 3-311 through 3-319.
K. Jack, Video Demystified: A Handbook for the Digital Engineer, Brooktree Corporation (1993), Ch. 6.

ART-UNIT: 277

PRIMARY-EXAMINER: Mai; Tan V.

ATTY-AGENT-FIRM: Blakely, Sokoloff, Taylor & Zafman

ABSTRACT:

The invention provides a method and apparatus for performing complex digital filters. According to one aspect of the invention, a method for performing a complex digital filter is described. The complex digital filter is performed using a set of data samples and a set of complex coefficients. In addition, the complex digital filter is performed using a inner and outer loop. The outer loop steps through a number of corresponding relationships between the set of complex coefficients and the set of data samples. The inner loop steps thorough each complex coefficient in the set of complex coefficients. Within the inner loop, the data sample corresponding to the current complex coefficient (the complex coefficient currently identified by the inner loop) is determined according to the current corresponding relationship (the corresponding relationship currently identified by the outer loop). Then, in response to receiving an instruction, eight data elements are read and used to generate a currently calculated complex number. These eight data elements were previously stored as packed data and include two representations of each of the components of the current complex coefficient and its current corresponding data sample. Each of these data elements is either the positive or negative of the component they represent. As a result of the manner in which these eight data elements are stored, the currently calculated complex number represents the product of the current complex coefficient and its current corresponding data sample. The currently calculated complex number is then added to the current output packed data.

14 Claims, 16 Drawing figures

L9: Entry 13 of 36 File: USPT May 22, 2001

DOCUMENT-IDENTIFIER: US 6237016 B1
**** See image for Certificate of Correction ****
TITLE: Method and apparatus for multiplying and accumulating data samples and complex coefficients

Detailed Description Text (9):
The decode unit 140 is shown including packed data instruction set 145 for performing operations on packed data. In one embodiment, the packed data instruction set 145 includes the following instructions: a packed multiply-add instruction(s) (PMADD) 150, a pack instruction(s) (PACK) 155, an unpack/interleave instruction(s) (PUNPCK) 160, a packed shift instruction(s) 165, an PXOR instruction(s) (PXOR) 170, a packed add instruction(s) (PADD) 175, a packed subtract instructions) (PSUB) 180, and a move instruction(s) 185. The operation of each of these instructions is further described herein. While these packed data instructions can be implemented to perform any number of different operations, in one embodiment these packed data instructions are those described in "A Set of Instructions for Operating on Packed Data," filed on Aug. 31, 1995, Ser. No. 08/521,360. Furthermore, in one embodiment, the processor 105 is a pipelined processor (e.g., the Pentium processor) capable of completing one or more of these packed data instructions per clock cycle (ignoring any data dependencies and pipeline freezes). In addition to the packed data instructions, processor 105 can include new instructions and/or instructions similar to or the same as those found in existing general purpose processors. For example, in one embodiment the processor 105 supports an instruction set which is compatible with the Intel Architecture instruction set used by existing processors, such as the Pentium processor. Alternative embodiments of the invention may contain more or less, as well as different, packed data instructions and still utilize the teachings of the invention.

Detailed Description Text (31):
FIG. 5 illustrates the operation of the unpack instruction according to one embodiment of the invention. In one embodiment, the unpack instruction interleaves the low-order data elements from a first operand 510 and a second operand 520. The numbers inside each packed data item identifies the data elements for purposes of illustration. Thus, data element 0 of the first operand 510 is stored as data element 0 of a result 530. Data element 0 of the second operand 520 is stored as data element 1 of the result 530. Data element 1 of the first operand 510 is stored as data element 2 of the result 530 and so forth, until all data elements of the result 530 store data elements from either the first operand 510 or the second operand 520. The high-order data elements of both the first and second operand are ignored. By choosing either the first operand 510 or the second operand 520 to be all zeroes, the unpack may be used to unpack packed byte data elements into packed word data elements, or to unpack packed word data elements into packed dword data elements, etc. In an alternate embodiment, the high-order bytes of each packed data item are interleaved into the result.

US Reference Patent Number (40):
5896543

WEST

Generate Collection Print

L12: Entry 9 of 18 File: USPT Jun 1, 1999

US-PAT-NO: 5909572
DOCUMENT-IDENTIFIER: US 5909572 A

TITLE: System and method for conditionally moving an operand from a source register to a destination register

DATE-ISSUED: June 1, 1999

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Thayer; John S.	Houston	TX		
Favor; John G.	Scotts Valley	CA		
Weber; Frederick D.	San Jose	CA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Compaq Computer Corp.	Houston	TX			02
Advanced Micro Device, Inc.	Sunnyvale	CA			02

APPL-NO: 08/ 759025 [PALM]
DATE FILED: December 2, 1996

INT-CL: [06] G06 F 9/00

US-CL-ISSUED: 395/567; 395/564, 395/566
US-CL-CURRENT: 712/226; 712/223, 712/225

FIELD-OF-SEARCH: 395/567, 395/564, 395/566, 395/562, 395/568, 395/563, 395/595, 395/800.35

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	3701977	October 1972	Mendelson et al.	711/126
☐	4707800	November 1987	Montrone et al.	364/788
☐	4725973	February 1988	Matsuura et al.	364/736.03
☐	4782441	November 1988	Inagami et al.	395/800.09
☐	4849882	July 1989	Aoyama et al.	395/800.06
☐	4884197	November 1989	Sachs et al.	711/123
☐	4891754	January 1990	Boreland	395/586
☐	5025407	June 1991	Gulley et al.	364/748.2
☐	5073864	December 1991	Methvin et al.	364/715.11
☐	5181183	January 1993	Miyazaki	364/725.03
☐	5187796	February 1993	Wang et al.	395/800.04
☐	5193167	March 1993	Sites et al.	711/163
☐	5226171	July 1993	Hall et al.	395/800.09
☐	5235536	August 1993	Matsubishi et al.	364/736.04
☐	5251323	October 1993	Isobe	395/800.05
☐	5307300	April 1994	Komoto et al.	364/736.01
☐	5335330	August 1994	Inoue	395/588
☐	5434592	July 1995	Dinwiddie, Jr. et al.	345/133
☐	5437043	July 1995	Fujii et al.	395/800.1
☐	5453945	September 1995	Tucker	364/725.01
☐	5471637	November 1995	Pawlowski et al.	394/296
☐	5511219	April 1996	Shimony et al.	395/800.35
☐	5513366	April 1996	Agrawal et al.	395/800.22
☐	5537640	July 1996	Pawlowski et al.	711/146
☐	5627981	May 1997	Adler et al.	395/582
☐	5640588	June 1997	Vegesna et al.	395/800.23
☐	5644520	July 1997	Pan et al.	364/736.01
☐	5655096	August 1997	Branigin	395/376
☐	5669013	September 1997	Watanabe et al.	395/825
☐	5673408	September 1997	Shebanow et al.	395/392
☐	5673426	September 1997	Shen et al.	395/591
☐	5692211	November 1997	Gulick et al.	395/800.35
☐	5745721	April 1998	Beard et al.	395/384
☐	5801975	September 1998	Thayer et al.	364/725
☐	5850227	December 1998	Longhenry et al.	345/439

OTHER PUBLICATIONS

A comparison of full and partial predicated execution support for ILP processors by Mahlke et al., 1995 IEEE publication, pp. 138-149.
Kohn, L, et al., "The Visual Instruction Set (VIS) in UltraSPARC," SPARC Technology Business--Sun Microsystems, Inc., 1996 IEEE pp. 462-489.
Gwennap, Linley, "UlatraSparc Adds Multimedia Instructions--Other New Instructions Handle Unaligned and Little-Endian Data," Microprocessor Report, Dec. 5, 1994, pp. 16-18.

Lee, Ruby B., "Realtime PMEG Video via Software Decompression on a PA-RISC Processor," Hewlett-Packard Company, 1995 IEEE, pp. 186-192.
Mattison, Phillip E., "Practical Digital Video With Programming Examples in C, " Wiley Professional Computing, pp. 158-178.
Zhou, Chang-Guo, et al., "MPEG Video Decoding With the UltraSPARC Visual Instruction Set," Sun Microsystems Inc., 1995 IEEE, pp. 470-474.

ART-UNIT: 278

PRIMARY-EXAMINER: Maung; Zarni

ATTY-AGENT-FIRM: Conley, Rose & Tayon Kowert; Robert C. Daffer; Kevin L.

ABSTRACT:

A multimedia extension unit (MEU) is provided for performing various multimedia-type operations. The MEU can be coupled either through a coprocessor bus or a local CPU bus to a conventional processor. The MEU employs vector registers, a vector ALU, and an operand routing unit (ORU) to perform a maximum number of the multimedia operations within as few instruction cycles as possible. Complex algorithms are readily performed by arranging operands upon the vector ALU in accordance with the desired algorithm flowgraph. The ORU aligns the operands within partitioned slots or sub-slots of the vector registers using vector instructions unique to the MEU. At the output of the ORU, operand pairs from vector source or destination registers can be easily routed and combined at the vector ALU. The vector instructions employ special load/store instructions in combination with numerous operational instructions to carry out concurrent multimedia operations on the aligned operands.

12 Claims, 19 Drawing figures

L12: Entry 9 of 18 File: USPT Jun 1, 1999

DOCUMENT-IDENTIFIER: US 5909572 A
TITLE: System and method for conditionally moving an operand from a source register to a destination register

Detailed Description Text (113):
FIG. 17 illustrates the vldb vd, mem128 and vstb mem64, vsh load/store instructions wherein 16 byte load and store operations occur in a 2:1 byte interleave pattern. A 10-bit load from memory address .alpha. maps the lower half of each slot s (i.e., lower half sub-slot) to the memory byte at address .alpha.+s; and it maps the upper half of each slot (i.e., upper half sub-slot) to the memory byte at address .alpha.+s+8. As a result, the MEU performs independent but identical operations on two sets of data that reside in two adjacent 8 byte octets of memory.

Detailed Description Text (116):
The interleave mapping for 10-bit partitions is completely transparent to the programmer as long as only 10-bit loads/stores and vector instructions are performed on a given set of data. Interleaved mapping of 20-bit partitions is also transparent to the programmer if only 20-bit operations are performed. However, if 10-bit and 20-bit operations are mixed, then care must be taken to understand the mapping so that the expected results are produced. The interleaving can be very useful, for example, if a 10-bit load from an octet-sized memory location automatically expands and interleaves the byte-wide memory data to the upper portion of 20-bit partitions. The 20-bit operation can be immediately performed on this data without the need for explicit format conversions. Subsequently, 10-bit stores to octets can automatically perform the inverse 20-bit to 10-bit packing function. Thus, the present store operation, namely vstb mem64, vsh performs packing of n+4 bits within a slot of a vector register to n/2 bits within an address of the memory unit. Given n=16, 20-bit-to-8-bit packing can occur as part of the store operation. Additional operations, such as move or shift operations need not occur to perform a packing function. Packing serves to store the most significant bits from a slot. Unpacking is an operation by which n/2 bits from a memory address are loaded into n+4 bit locations within a slot. If n=16, then a load operation such as vldb vdh, mem64 causes 8-bits within a memory address to be loaded into a 20-bit slot. Utilizing load and store functions in such a manner thereby avoids having to implement separate unpack and pack instructions, respectively, within the MEU instruction set. Accordingly, the same result can be achieved but with fewer instructions. For MPEG, 8-bit pixels are unpacked to 20-bit numbers for DCT or IDCT manipulations, then the results are repacked to 8-bit pixels. The internals of the DCT and IDCT operations require more than 8 bits of precision, to which packing and unpacking are particularly advantageous.

Detailed Description Text (119):
Movement of data not only between slots, but between sub-slots is particularly helpful when performing MPEG motion compensation on 8-bit pixel values. In the example shown above, a single load instruction which causes interleaving of 16-bytes, followed by four move and four sub-slot routing instructions performs the same function but in a more efficient manner than doing unaligned memory references. Thus, MPEG motion compensation on a 1.times.8 block is advantageously performed by a single interleaving load operation, followed by a single vector instruction containing three move operations (mov) and five sub-slot swapping operations (blbh) across five slot midpoints.

Detailed Description Paragraph Table (7):

```
_____ ;16 video bytes are in data in memory (the MSB,
A, is shown on left): ;ABCD EFGH IJKL MNOP ;need to extract 8 unaligned bytes from
center; FCHI JKLM ;load 16 bytes into register v0 (load does interleaving) vldb v0,
byte ptr [esi] ;esi points to byte "P" ;now v0 contains AIBJ CKDL EMFN GOHP ;in slots:
7766 5544 3322 1100 ;use 20-bit routing ops to move data across 10-bit routing barrier
{mov mov mov blbh blbh blbh blbh blbh} word v0, v0, v0(21076543) ;now v0 contains FNGO
HPIA JBKC LDME = FxGx HxIx JxKx LxMx ;store 8 bytes into memory vstb byte ptr [edi],
v0h ;*[edi] contains FGHI JKLM _____
```

WEST

Generate Collection | Print

L3: Entry 1 of 3 File: USPT Jul 10, 2001

US-PAT-NO: 6260137
DOCUMENT-IDENTIFIER: US 6260137 B1

TITLE: Data processing unit with digital signal processing capabilities

DATE-ISSUED: July 10, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Fleck; Rod G.	Mountain View	CA		
Martin; Daniel	Mountain View	CA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Siemens Aktiengesellschaft	Munich			DE	03

APPL-NO: 08/ 928764 [PALM]
DATE FILED: September 12, 1997

INT-CL: [07] G06 F 9/315

US-CL-ISSUED: 712/225; 712/224
US-CL-CURRENT: 712/225; 712/224

FIELD-OF-SEARCH: 395/393, 395/588, 395/598, 395/800.13, 395/800.22, 395/800.23, 395/800.24, 395/800.35, 395/800.36, 395/800.41, 395/380, 395/306, 395/316, 712/217, 712/241, 712/248, 712/1-43, 712/2T, 712/224, 712/225, 710/9, 711/1-6, 711/1T

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected | Search ALL

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	4992934	February 1991	Portanova et al.	712/209
☐	5269007	December 1993	Hanawa	712/218
☐	5367705	November 1994	Sites	710/21
☐	5574928	November 1996	White	712/23
☐	5721892	February 1998	Peleg	712/221
☐	5734874	March 1998	Van Hook	345/513
☐	5752271	May 1998	Yung	712/23
☐	5768609	June 1998	Gove et al.	712/11
☐	5778241	July 1998	Bindloss et al.	712/20
☐	5812147	September 1998	Van Hook	345/511
☐	5852726	December 1998	Lin et al.	712/200
☐	5864713	January 1999	Terry	395/872
☐	5896543	April 1999	Garde	712/35
☐	5913054	June 1999	Mallick et al.	711/200
☐	5918252	June 1999	Chen et al.	711/217
☐	5983256	November 1999	Peleg et al.	708/523

FOREIGN PATENT DOCUMENTS

FOREIGN-PAT-NO	PUBN-DATE	COUNTRY	US-CL
0 473 805 A1	September 1990	EP	
0 679 991 A1	November 1995	EP	
WO 96/17291	June 1996	WO	

OTHER PUBLICATIONS

Sun Microsystems, "Ultrasparc-I,-II User's manual", pp. 190-234, [retrieved on Jul. 7, 1999]. Retieved from the Internet:<URL: http://www.sun.com/microelectronics/UltraSPARC-II/; $sessionid$2EHV1ZQAAFGW5AMUVFZE5YQ.*
Sun Microsystems, "Ultrasparc-IIi User's manual", pp.127-133, Oct. 1997.

ART-UNIT: 282

PRIMARY-EXAMINER: Niebling; John F.

ASSISTANT-EXAMINER: Whitmore; Stacy

ABSTRACT:

The present invention relates to a data processing unit comprising a register file, a register load and store buffer connected to the register file, a single memory, and a bus having at least first and second word lines to form a double word wide bus coupling the register load and store buffer with said single memory. The register file at least two sets of registers whereby the first set of registers can be coupled with one of the word lines and the second set of registers can be coupled with the respective other word lines, a load and store control unit for transferring data from or to the memory.

24 Claims, 9 Drawing figures